Evaluation of Cross-Layer Reliability Mechanisms for Satellite Digital Multimedia Broadcast

Amine Bouabdallah, Michel Kieffer *Member, IEEE*, Jérôme Lacan, Galina Sabeva, Gaël Scot, Caroline Bazile, Pierre Duhamel *Fellow, IEEE* .

Abstract

This paper presents a study of some reliability mechanisms which may be put at work in the context of Satellite Digital Multimedia Broadcasting (SDMB) to mobile devices such as handheld phones. These mechanisms include error correcting codes, interleaving at the physical layer, erasure codes at intermediate layers and error concealment on the video decoder. The evaluation is made on a realistic satellite channel and takes into account practical constraints such as the maximum zapping time and the user mobility at several speeds. The evaluation is done by simulating different scenarii with complete protocol stacks. The simulations indicate that, under the assumptions taken here, the scenario using highly compressed video protected by erasure codes at intermediate layers seems to be the best solution on this kind of channel.

Index Terms

Broadcasting, Channel coding, Multimedia communication, Video coding.

This work was partly supported by the CNES SDMB project.

Amine Bouabdallah and Jérôme Lacan are with ENSICA/TéSA and with LAAS/CNRS, 1, place E. Blouin, 31056 Toulouse Cedex, Toulouse, France (e-mail : {amine.bouabdallah, jerome.lacan}@ensica.fr). Michel Kieffer, Galina Sabeva and Pierre Duhamel are with LSS (CNRS, Supélec, Univ Paris-Sud), F-91192 Gif-sur-Yvette, France (e-mail: {michel.kieffer, galina.sabeva, pierre.duhamel}@ lss.supelec.fr). Gaël Scot and Caroline Bazile are with CNES, 18 avenue Edouard Belin - F-31401 Toulouse Cedex 4, France, (e-mail : {gael.scot,caroline.bazile}@cnes.fr)

I. INTRODUCTION

There is currently a growing demand for good-quality and location-independent access to multimedia contents. Applications such as video on demand or TV on mobile devices such as handheld phones, PDA, or notebooks should develop widely in the near future. Solutions based on radio access networks (RAN) such as GPRS, UMTS, or CDMA–2000 are well-adapted for unicast communications but not viable for the long term, since such networks are not likely to support, *e.g.*, massive consultations of several TV channels in parallel.

The third-generation partnership project (3GPP) is currently investigating Multimedia Broadcast/Multicast Services (MBMS) extensions to these 2G and 3G RAN [1]. Nevertheless, MBMS seems currently still not suited for the broadcasting of many TV channels over a wide area. Satellite Digital Multimedia Broadcasting (SDMB) represents an interesting proposal to respond to this demand, as well as some other alternatives such as Terrestrial DMB [2] and Digital Video Broadcasting-Handheld (DVB–H) [3]. SDMB is especially interesting to ensure coverage of rural and small towns environments, for which it provides moderate deployment costs, when compared to T-DMB or DVB-H.

In this context, the SDMB project conducted by CNES, connected with the Unlimited Mobile TV project of Alcatel, aims at delivering live TV channels directly to mobile handheld phones through an hybrid infrastructure composed of a satellite offering a broad coverage of a large area and terrestrial repeaters for urban complementary coverage (see Figure 1). The objective is to broadcast QVGA or CIF encoded video with a typical throughput of about 250 kbps per channel.

Typical issues that have to be addressed in such SDMB application are in part similar to those encountered in any wireless multimedia transmission context, see [4]. The characteristics of the wireless channel are location-dependent and thus time-varying when the receiver moves. The receiver has to comply with signals varying in a large dynamical range, from good conditions, when the satellite or a repeater is in Line Of Sight (LOS), to quasi-loss of the received signal, when deep satellite shadowing occurs out of repeater coverage [5]. Automatic Repeat reQuest (ARQ) mechanisms are very difficult to implement in SDMB, mainly due to the potentially high number of receivers. The transmitter has thus to provide a signal which may be well decoded by the largest proportion of receivers. Transmission delays are less critical than in conversational

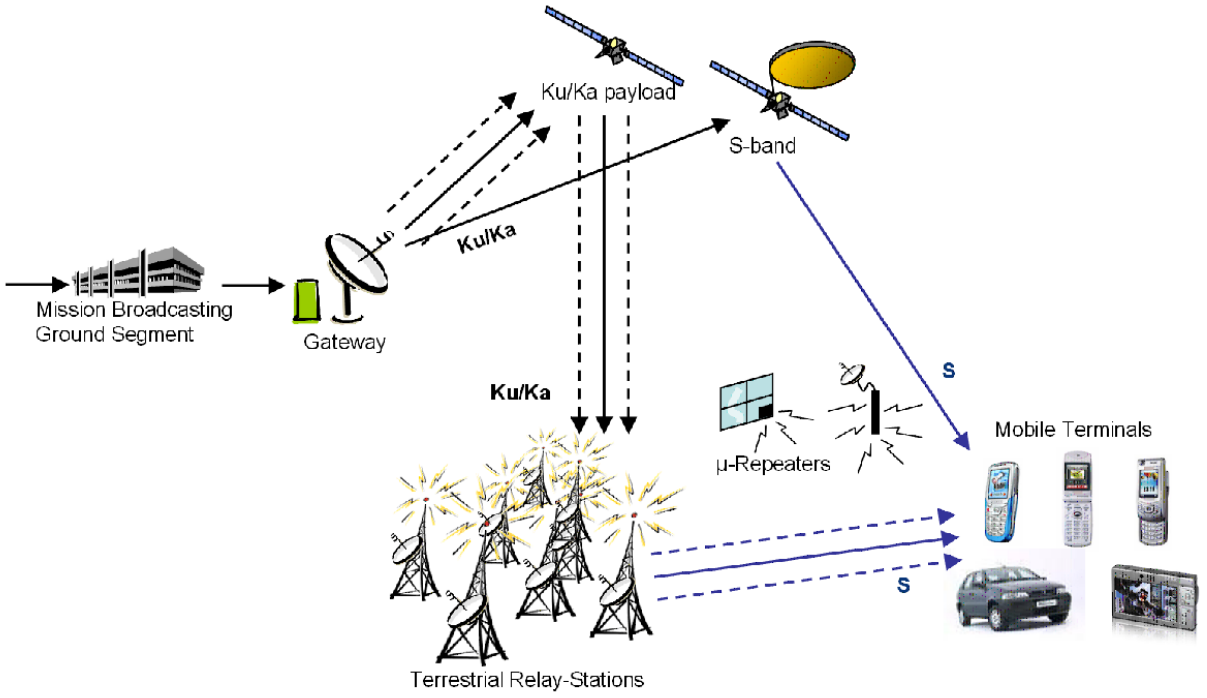

Fig. 1. Representation of the SDMB system

applications. More important is the zapping time, *i.e.*, the time necessary to switch between two channels, which has to remain as small as possible and is a specific constraint of DMB.

Redundancy introduction and error concealment are the tools of choice for addressing these issues. In [4], several application-layer mechanisms, parts of the H.264/AVC standard [6], are presented to provide efficient video transmission over wireless error-prone channels. In the context of SDMB, the satellite propagation channel is such that redundancy has also to be introduced at lower protocol layers in order to cope with fading and possible deep shadowing. The optimization of the allocation of redundancy between physical, intermediate and application layers is then a natural question which arises in this context [7]. Nevertheless, the absence of feedback channel in the context of SDMB imposes a worst-case optimization in the place of continuous adaptations suggested by [7]. These questions have also been considered in [8] in the context of MBMS.

The aim of this paper is to present a solution for SDMB combining redundancy introduction at

the physical, network, and application layers. Redundancy introduction and error concealment at the application layer concur to mitigate the effects of transmission channel variations, allowing good video quality for a given bit-budget with a reasonable zapping time. Performance evaluations are done on a realistic satellite channel model. Some interactions between the different protocol layers will be illustrated, evidencing the need for a cross-layer optimisation of the robustness introduction. More sophisticated tools such as robust decoders taking into account the redundancy left within the encoded video stream [9], [10] will not be considered here. Possible places where these techniques may be helpful are pointed out.

The channel model presented in Section II evidences the previously mentioned difficulties which have to be faced in the context of SDMB. The proposed protocol stack and system architecture are then detailed in Section III. The various techniques involved in the protocol stack to increase the robustness of the compressed video to variations of the channel characteristics are presented in Section IV. Section V provides some simulation results for the whole system, before drawing some conclusions.

II. CHANNEL MODEL

The satellite transmission channel model largely determines the design and dimensioning of the different layers of the protocol stack employed for SDMB. In this paper we focus on the case of satellite-only reception. This situation represents a worst-case study, as repeaters may reduce the occurrence of loss of signals.

The satellite link budget is the most important constraint in the design of a SDMB system. It determines the received signal-to-noise ratio in LOS. Usually, shadowing and blocking phenomena on the satellite link have a low coherence distance. An accurate determination of this distance is useful for dimensioning interleavers at the physical layer or at intermediate layers. Contrary to propagation channel of repeaters, the satellite channel is usually considered as flat in the signal bandwidth of 5 MHz. To cope with frequency selectivity of the transmission channel in repeaters areas, selection of adapted waveforms such as OFDM is necessary. On satellite link, different physical layer options are possible depending on the frequency band used. For a system privileging frequency efficiency by the use of the same frequency band for both satellite and repeater links, the same type of OFDM signal may be used. If the two links use separate bands, a better optimisation of satellite link is possible and the use of a more classical TDM signal can

be preferred for its better resistance to non-linear satellite amplification.

The propagation part of the channel model considered in the paper is a three-states model (LOS/Shadowing/Blockage) based on Markov chains integrating three scales of propagation effects [11], [5]. The large scale effects correspond to the changes between the three states. The mid scale effect or "Slow" fading corresponds to shadowing. The small scale effect or "Fast" fading is due to multipath. Based on extensive measurement campaigns, [5] defines transition matrices between the three states for five environments (Open, Intermediate tree shadowed, Heavy tree shadowed, Suburban, and Urban), for elevation angles above 40°. The environment selected in this paper is the Intermediate tree shadowed, which is rather severe in terms of shadowing attenuations compared with the open and even with the Suburban cases. For each state, slow fading and fast fading variations are defined according to a Loo distribution [12]. The Loo distribution considers that the received signal is made up of the sum of two components, the direct signal and multipaths. Within one state, the dynamics of the shadowing affecting the direct path is given by its correlation distance, equal in Intermediate tree shadowed environment to 1.5 m.

Moreover, a perfect demodulation is assumed. Simulations are done with a TDM signal, which is expected to be representative for both TDM and OFDM cases. Indeed, for OFDM signals, the symbol duration is equal to a few hundreds of micro-seconds, which corresponds to a rather stable propagation attenuation even for the highest speed considered. Noise level adjustment compared with LOS signal level is based on the assumption of a margin of 10 dB, which corresponds to a typical link budget, as detailed in [13].

The receiver carrier-to-noise ratio (C/N) is simulated over a distance of 1 km, in order to limit the duration of the simulations performed in the upper layers. The simulator accounts for the modulation, the propagation part, and the demodulation. The considered simulation distance is split into 220000 segments of 0.004545 m each. The simulated data, represented in Figure 2, cannot be considered as perfectly representative for the complete environment but give a good idea of the channel behavior. The distance may then be covered at various speeds.

As can be seen on Figure 2, the received carrier-to-noise ratio ranges from approximatively 10 dB to less than -25 dB during deep shadowing. Moreover, deep shadowing effects appear sometimes over distances larger than 50 m. Such situations are particularly challenging, especially when the receiver moves slowly.

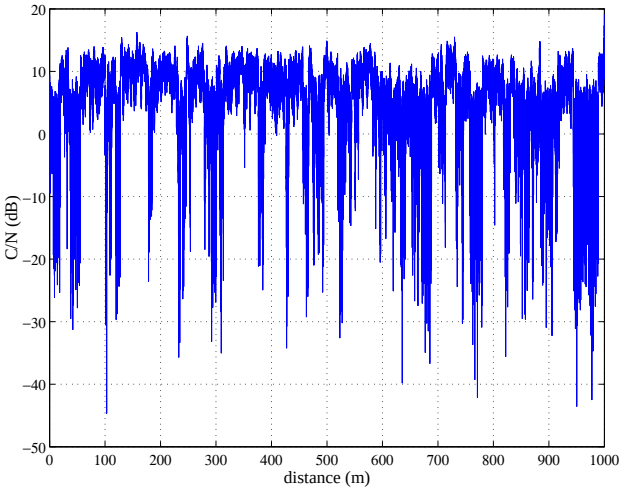

Fig. 2. Evolution over 1 km of the receiver signal-to-noise ratio for the Intermediate tree shadowed scenario

III. Protocol stack for SDMB

The considered protocol stack, represented on Figure 3, has been largely inspired from DVB–H [3], in order to improve interoperability with this broadcast technique and minimize terminal chipset modifications. This choice is well-suited for a signal which may be received either directly from the satellite or from a repeater. However, modifications of the DVB–H stack are proposed in order to improve the robustness to fading and shadowing of SDMB.

The next sections detail the protocol layers of Figure 3. The reliability mechanisms which may be integrated on these layers are considered in Section IV.

A. Application layer

The considered video coder, H.264 [6], consists of two parts. The Video Coding Layer (VCL) is responsible for the video compression. The Network Abstraction Layer (NAL) performs the interface between the VCL and the lower protocol layers. The NAL produces NAL Units (NALUs) which may be concatenated in order to form a bitstream suitable for storage or encapsulation into Real-time Transport Protocol (RTP) [14] packets according to the RTP payload specification for H.264 [15].

The VCL divides each frame n of the video sequence into *macroblocks* of 16×16 pixels. A group of macroblocks forms a *slice* of the frame. Each slice is encoded independently. For

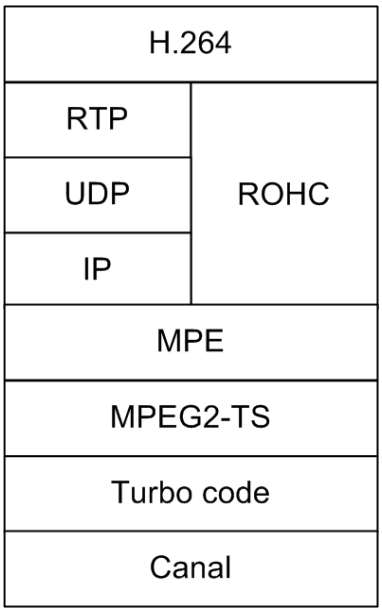

Fig. 3. Considered protocol stack

the m-th macroblock $M_{n,m}$ of a slice, a prediction $P_{n,m}$ is evaluated and subtracted from the original macroblock $M_{n,m}$. The result is referred to as a *difference* macroblock $D_{n,m}$ and is passed through a video processing module, performing discrete cosine transform, quantization and finally entropy coding. There are two types of prediction: *intra* and *inter*. For intra prediction, the encoder uses only information from the current slice. For inter prediction, data from the other encoded frames may also be used. Inter-encoded frames may be of types P and B. For P frames, only the previously encoded frames are used for prediction, whereas for B frames, bidirectional prediction is performed. Usually, B frames do not serve as reference. Within P or B frames, macroblocks may be encoded in *intra* mode.

The most common NALU organisation is to place one slice per NALU. Loosing a NALU, except when it corresponds to a slice of a B frame results in errors in the corresponding frame, but also in the frames which use it as reference for prediction. The probability of loosing a NALU mainly depends of the size of the NALU and of the packets provided by the lower protocol layers. The optimization of the size of the NALU will be discussed in Section IV-A.

B. Intermediate Layers

The convergence of the different network technologies in the next few years is now established. For this reason, the main next generation networks are designed to transport IP data. In this context, the DVB consortium has introduced the IP Datacast architecture [16] defining the protocol stacks to transmit IP data over DVB-H. The proposed Transport and Network layers comply with this architecture.

1) Transport and Network layers: As seen in Section III-A, RTP packets are provided by the NAL of H.264. Even if it is possible to encapsulate several NALUs in a single RTP packet, in the context of this work where relatively high packet loss rate may be encountered, the encapsulation of a single NALU per RTP packet is more efficient. The main function of RTP is to add a timestamp, a sequence number and to identify the payload type. The length of the RTP header (with only fixed fields) is 16 bytes. Note that SDMB does not consider RTCP [14], which would require a feedback channel not realistic for satellite broadcast.

Since SDMB considers unidirectional broadcast, the transport layer is not in charge of the reliability nor the congestion control and UDP [17] is used. The length of the UDP header is 8 bytes, including the sender and receiver port and a 16-bit checksum detecting errors on the whole packet.

The network protocol is IP, two versions of which, IPv4 [18] and IPv6 [19] may be used. Their header are respectively 20 and 40 bytes long.

The RTP, UDP, and IP encapsulations add thus headers of 44 bytes (with IPv4) and 64 bytes (with IPv6) per NALU. As will be seen in Section IV-A.1, the desirable length of the NALU in the context of SDMB is between 100 and 400 bytes. The headers thus represent an extremely large proportion of the IP packets. For this reason, the use of header compression mechanisms is recommended. For the protocols used here, ROHC [20] is the best candidate since it is able to consider together the RTP, UDP and IP headers in the compression process. Initially designed for bi-directional links, ROHC integrates an unidirectional mode that may also be used in the present broadcasting context, as detailed in [21].

In the simulations, we chose to implement ROHC instead of RTP, UDP and IP. Since it is very difficult to estimate the average length of the ROHC header in unidirectional links, we fixed the header length to 20 bytes (estimated from the results of [21]). In practice, this header is directly added to the NALUs provided by the H.264 video codec. At the decoder side, this header is

simply erased.

2) Link layer: Since the MPEG2 Transport Stream (MPEG2 TS) packets [22] are the data units handled by most of the DVB physical layers, the adaptation layer must be used to map variable-length IP packets onto fixed-length MPEG2 packets. Even if the ULE protocol [23] may be used, DVB recommends the use of Multi-Protocol Encapsulation (MPE) [24]. The header added by MPE to each IP packet is 12 bytes long, a 4 bytes long CRC is also appended. For transmissions toward handheld devices, two optional mechanisms are proposed: time-slicing and MPE-FEC [24].

The concept of time slicing is to send data in bursts. Between each burst, data of the elementary stream are not transmitted, allowing other elementary streams to use the bandwidth otherwise allocated. This enables a receiver to stay active only for a fragment of the time, while receiving bursts of the service it requests. Time slicing thus limits the receiver power consumption. The structure of the MPE-FEC erasure code is described in Section IV-B.

The aim of MPEG2 is to multiplex several sources of compressed data. As already explained, the packets provided by MPE are organized into MPEG2 packets of 188 bytes. Among the 188 bytes, 4 are dedicated to the packet header.

In the simulations, MPE uses the section-packing algorithm to build MPEG2 packets, *i.e.*, it allows MPEG2 packets to carry parts of several MPE packets. The optional time-slicing was not implemented since the power consumption is not one of the main objectives of this work. The second optional mechanism, MPE-FEC, was not used either, because an erasure code at transport level was preferred (see the justification in Section IV-B).

C. Physical layer

The modulation and channel model used by SDMB has been described in Section II. Here, we focus on the construction of the transport stream and on the channel code of the physical layer.

In DVB–H, which uses the same channel coding as DVB–T [25], convolutional codes are concatenated with Reed-Solomon codes. Several solutions may be considered to improve the decoding efficiency and thus increase the throughput of the physical layer for a fixed link budget. One may consider turbo codes, such as those proposed by the 3GPP for W-CDMA [26] or by the 3GPP2 for CDMA2000 [27]. LDPC codes [28] such as those adopted in DVB S2 [29] are

also a potential alternative. Moreover, the use of a long interleaving at physical layer may be considered to enhance the air interface performance and to mitigate satellite channel impairments. The increase of this physical layer interleaving has to be kept compatible with zapping time constraints.

In this paper, the rate $1/3$ turbo code specified by [26] is considered. It processes each 188-byte MPEG2 packet independently to generate a codeword of 564 bytes. The rate of the channel code is one of the constraints of the "SDMB" and "Unlimited Mobile TV" projects respectively conducted by CNES and Alcatel Alenia Space, and we did not consider other options. Nevertheless, the opportunity to use long interleavers at the physical layer will be discussed in Section V.

IV. Increasing robustness

At receiver side, a noisy version of the transmitted bitstream is fed to the turbo decoder. The output of the turbo decoder is put into MPEG2 packets which are delivered to the upper protocol layers. In the proposed scheme, as Reed-Solomon codes concatenated with convolutional codes have been replaced by turbo codes, it is not easy to determine whether an MPEG2 packet is erroneous, see [30].

The MPE layer implements a CRC protecting the whole MPE sections to detect residual errors. Any MPE section detected as erroneous is discarded. The error channel is then transformed into a packet erasure channel. The aim of this section is to describe several robustness tools to recover or conceal lost packets.

A. Application layer

At application layer, the redundancy introduction and error-concealment mechanisms put at work are the same as those used in wireless conversational applications. They are briefly recalled here. For more details, see [4].

1) Optimizing the size of the NALU: H.264 does not allow to get NALUs, and thus RTP packets, with a constant size. Nevertheless, using slice-structured coding [6], it is possible to limit the size of each NALU. This may be helpful for controlling the proportion of lost RTP packets due to erroneously received MPEG2 packets.

A lost MPEG2 packet may result in the loss of one to several NALUs, depending of the size of the NALUs. When large NALUs are considered, the loss of a small MPEG2 packet may result in the loss of a large amount of data. From a NALU-loss point of view, using small NALUs is thus recommended. However, as NALUs are encoded independently, the probability counts of the adaptive entropy coders of H.264 are reinitialized at the beginning of each NALU, which is detrimental for the coding efficiency. Moreover, small NALUs result in RTP packets with poor ratios payload over total packet size. The size limit of the NALUs has thus to be carefully optimized. Note that the problem of optimization of the application-layer packet size as a function of the channel condition has been previously considered in [31] in the context of video transmission over 802.11 WLANs. Here, it will be studied as a function of the MPEG2 packet loss rate.

A sequence of NALUs is randomly generated according to a gaussian probability distribution with mean μ and variance $\sigma^2 = 100$, truncated at $\mu + 10$ to account for the size limit imposed by H.264. As seen in Section III-B, the average length of the ROHC headers is estimated to 20 bytes. Additionally, the header and CRC of the MPE layer have a total length of 16 bytes. Thus headers of 36 bytes are appended to the generated NALUs. The resulting packets are then mapped to MPEG2 packets. Packet loss rates of 5%, 10%, and 15% are considered. Figure 4 gives the goodput, *i.e.*, the ratio between the number of correctly received information bits (headers and CRC have been removed) and the number of bits sent into MPEG2 packets as a function of the packet-size limit ($\mu + 10$). Each point is simulated with bitstreams of 2.10^4 NALUs.

The optimal NALU size limit depends on the MPEG2 packet-loss rate. For 15% of lost packets, a NALU size limit of about 190 bytes provides the best goodput. For a loss rate of 10%, NALU size limits between 150 and 350 bytes perform almost similarly, whereas for 5% of lost packets, NALUs of more than 200 bytes may be used.

The previous study does not consider the loss in coding efficiency when the NALU gets smaller. To study the effect of size limits, the JM 10.2 implementation of H.264 in its high profile has been used. The frames of the `foreman.cif` video sequence are encoded at 15 frames per second (fps) according to the following scheme: IBPBP... An intra-coded (I) frame is inserted every 16 frames. The quantization parameters for each frame type are $Q_{\mathrm{I}} = 36$, $Q_{\mathrm{P}} = 34$, and $Q_{\mathrm{B}} = 30$. Slice-structured encoding is performed with NALUs limited to 120, 160, and 180 bytes. The reference is the unlimited case, in which a NALU consists of a whole

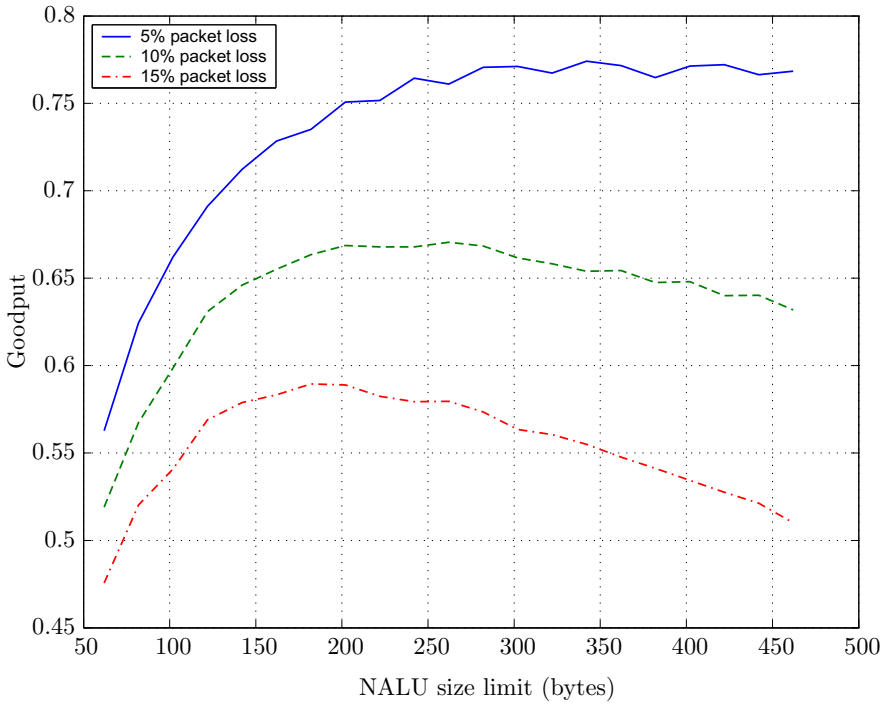

Fig. 4. Proportion of transmitted information bits for various NALU size limit

encoded frame. The evolution of the size of the bitstream generated by the H.264 video coder is illustrated by Table I. Even for very small NALU (120 bytes), the redundancy is relatively small (10.6%). The conclusions obtained from Figure 4 are thus unchanged. Nevertheless, an optimisation of the NALU size limit taking into account the effect of concealment performed by the video decoder would provide more accurate results.

NALU size	SNR-Y(dB)	Bitstream size	Extra bits
frame	33,87	2176384 bits	
180 bytes	33,82	2328976 bits	9640 (6.8%)
160 bytes	33,83	2338320 bits	19240 (7.4%)
120 bytes	33,82	2407712 bits	52776 (10.6%)

TABLE I

REDUNDANCY INTRODUCED WHEN LIMITING THE SIZE OF NALUS (FIRST 197 FRAMES OF FOREMAN.CIF)

2) Error concealment: When a NALU has been lost, error concealment techniques may be used to limit the visual impact of the lost slices on the reconstructed video stream. Most concealment techniques rely on the hypothesis that the video sequence is spatially and temporally correlated. Good reviews of such techniques may be found in [32], [33].

Two concealment techniques have been implemented in the JM 10.2 of H.264. First, all received NALUs corresponding to a given frame are decoded. All macroblocks that were not decoded due to losses of NALU are marked and error concealment is performed. A first concealment technique consists in estimating the pixels of a lost macroblock in such a way that they are as close as possible to the pixels of their neighboring macroblocs, as suggested by [34]. An alternative approach, suited for inter-encoded frames consists in estimating the motion vector of the macroblock to conceal with the motion vectors of the surrounding macroblocks and to use the previously decoded frames to perform motion reconstruction, as suggested by [35], [36].

Both concealment techniques are satisfying when there are only few consecutive lost slices and more generally when the lost macroblocks are spread on the frame. Picking the macroblocks randomly to fill a slice, as allowed by the Flexible Macroblock Ordering option of H.264, may help to improve the concealment efficiency.

B. Intermediate layers: Erasure codes

On the packet erasure channel seen by the intermediate layers, erasure codes can be used at several levels. First, since MPE has defined his own MPE-FEC erasure code [24], the link layer could be the right place to implement it. MPE-FEC is a Reed-Solomon erasure code of dimension 191 and length 255. Its input is a set of 191 columns of fixed-length obtained by segmenting the stream of the IP packets. It produces $255-191=64$ redundant columns that are considered as MPE packets. The number of rows of the matrix is a parameter determining the level of interleaving which is adapted to the context. It should be noted that the parameters of the erasure code can be modified by shortening and puncturing operations. The main drawback of this code is the limited maximum length. Indeed, the more the erasure code combines packets, the more efficient it is.

Longer codes, such as Raptor codes [37], LDGM codes [38], or simply longer Reed-Solomon erasure codes [39], which are candidates for standardization by DVB and/or IETF organisms, can be used. They are used between application and transport layers and directly work on the

data units produced by the application layer.

Since our goal is to evaluate the potential of the different mechanisms, Reed-Solomon erasure codes have been implemented between application and transport layers, which allows much more freedom on the length of the code. The parameter T_{block} represents the time needed to transmit all the packets belonging to a given codeword and thus determines the length of the code. For example, H.264 encoding a video at 150 kbps with NALUs limited to 120 bytes, produces roughly 180 NALUs per second. For $T_{\text{block}} = 1$, the erasure code builds then 120 redundant packets and the $180 + 120 = 300$ resulting packets constitute a codeword, see Figure 5. The length of each redundant packet is the size limit of the NALUs.

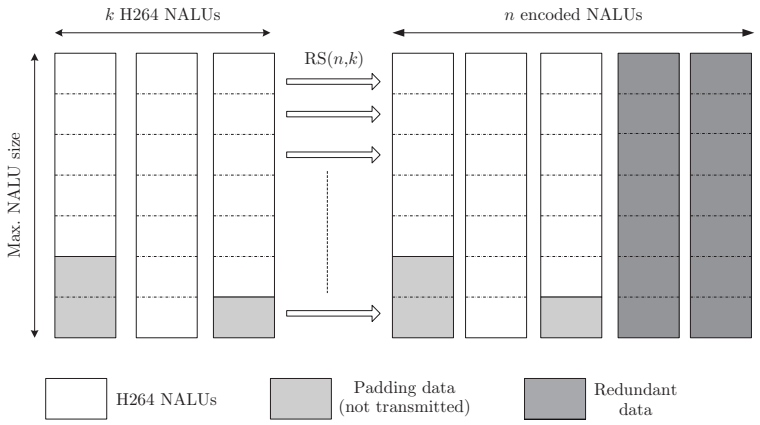

Fig. 5. Construction of the redundant packets using the Reed-Solomon erasure code

The performance of the erasure code strongly depends on the parameter T_{block} because a code with long codewords is less sensitive to bursts of lost packets. However, the choice of a high value of T_{block} has several negative consequences. The main drawback is that the lost packets can be recovered only when the whole block is received. Therefore, the real-time constraints of the application must be taken into account to the choice of the value of T_{block}. A strong constraint is the zapping time, which is between T_{block} and $2T_{\text{block}}$. Long codes also need more CPU and memory resource to encode and decode the messages. However, it has been shown in [40] that the CPU resource needed to decode the erasure code can be neglected compared to that used for video decoding, especially for erasure codes based on sparse matrices.

C. Physical layer: Interleaving

One of the main characteristics of the channel described in Section II is a strong spatial correlation (characterized by a correlation distance of 1.5 m). A direct consequence is that a moving receiver observes bursts of erroneous bits or of packet losses. A classical solution to cope with these bursts is to use interleaving or spreading techniques in order to distribute in time (and thus in space for a moving receiver) the erased packets or erroneous bits of the codewords.

The spreading consist in organizing the emission of a codeword in order to send its different parts in different time slots. In practice, spreading is often implemented through interleaving techniques which allow to send simultaneously several codewords in order to keep an equivalent throughput. Figure 6 illustrates the use of an interleaving of depth 2 (the spreading factor is also equal to 2). As shown in Figures 6 (b) and (c), the use of a spreading factor of 2 for a receiver

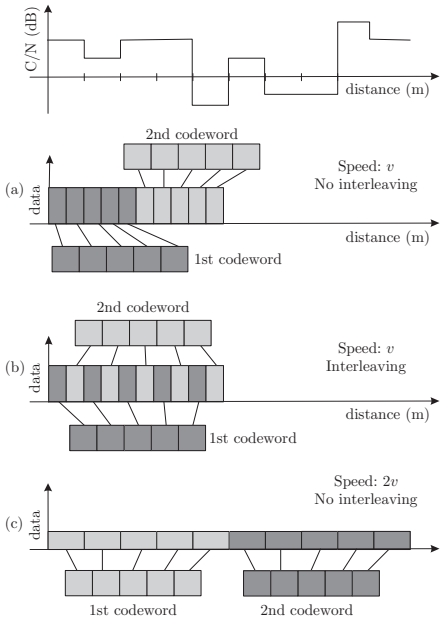

Fig. 6. Relation between (a) sequential transmission with mobile at speed v, (b) interleaved transmission with mobile at speed v and (c) sequential transmission with mobile with speed $2 * v$

with speed v leads to bursts equivalent to those observed by a receiver with speed $2v$ (without spreading). Actually, it can be observed that two receivers observe bursts with similar parameters if they have the same product *speed* $\times$ *spreading factor*. This observation is confirmed by the simulation results presented below.

The spreading with possible interleaving may be implemented at several layers. First, the DVB-H physical layer can mimic UMTS by distributing the data of a codeword into different time slots. Related to the bursty nature of the channel, this solution may be a good candidate to improve the global reliability, provided that the interleaving level complies with the limit on the zapping time.

Interleaving may also be used at MPE layer with MPE-FEC which encodes the rows of a matrix whose columns are filled by consecutive IP packets. If the columns are chosen sufficiently large, the FEC codes encodes data belonging to non consecutive IP packets.

However, it is well known that long codes are more efficient in terms of correction capability than interleaved short codes [41]. Thus, we choose to use long erasure codes between application and transport level instead of a short code and interleaving at MPE layer. Actually, the use of long codes, which combines a large number of packets (and thus packets spaced in the time), can be considered as a form of spreading.

V. SIMULATIONS

The protocol stack presented in Section III, serves as a reference for the simulations. The influence of the reliability mechanisms described in Section IV is evaluated with four scenarii. The section starts with a description of the common parts for all scenarii: the video coder and decoder and the description of simulations on the physical and intermediate layers.

A. Conditions

1) Video coder and decoder: For the simulations, the same video coder as in Section IV-A.1 is employed. The frame rate (15 fps), slide structure (IBPBP...), and frequency of I frames (1 every 16 frames) are also the same. This high frequency of I frames is necessary to limit the zapping time (here, about 1 s). Loops built with the 190 first frames of `foreman.cif` are encoded at a rate R between 100 kbps and 250 kbps depending on the scenario. The initial values of Q_{I}, Q_{P}, and Q_{B} are adjusted in such a way that there is no large variation of their value during the regulation of the rate at the beginning of encoding. Slice-structured encoding is enabled. NALUs are limited to 120 bytes in most of the cases, which allow an efficient error concealment, for a satisfying goodput at the cost of a reasonable redundancy introduction, see Section IV-A.1. Error concealment is activated at the decoder for all scenarii.

Speed (m/s)	BER	PLR	BERLP
1.5	3.13%	12.8%	24%
15	2.18%	19.6%	11%
31	1.52%	21.9%	8.0%

TABLE II

BIT ERROR RATE AT THE OUTPUT OF THE TURBO DECODER, PACKET LOSS RATE AT THE OUTPUT OF MPE, AND BET ERROR RATE IN THE ERRONEOUS MPE PACKETS.

At decoder output, to evaluate the image quality, the mean squared error σ_n^2 between the luminance of the n-th original frame and that of the reconstructed frame is evaluated. The PSNR for the n-th frame is then given by

$$\text{PSNR}_n = 10 \log_{10} \frac{255^2}{\sigma_n^2} \text{ (in dB)}.$$

The average mean squared error σ^2 is evaluated for the N frames of the video sequence and the average PSNR is

$$\text{PSNR} = 10 \log_{10} \frac{255^2}{\sigma^2} \text{ (in dB)}.$$

2) Intermediate layers: The CRC of the MPE packets is used to detect erroneous MPE packets which are consequences of noisy MPEG2 packets provided by the physical layer.

Table II shows the average bit error rates (BER), packet loss rates (PLR), and bit error rates in the lost packets (BERLP) as a function of the speed. Though the average output BER is lower for high speeds than for low speeds, the number of erroneous blocks after processing by MPE is larger for high speeds than for low speeds. For low speeds, each erroneous packet contains more errors than for high speeds. This can be explained by remembering that an increase of the speed plays a role similar to an increase of the interleaving depth, see Section IV-C. Increasing the speed improves the performance of the turbo decoder in terms of bit-error probability. Nevertheless, as long as the spreading is not efficient enough to obtain a vanishing error probability at the output of the turbo decoder, bit errors will remain present and will be distributed in more packets than without spreading, see [42] for more details. A further analysis of the influence of speed and interleaving is provided in Section V-C.

3) Common physical layer: Simulations for the modulation, the propagation part, and the demodulation are described in Section II. The bits forming the MPEG2 packets are encoded using the turbo code described in Section III-C. Each pair of encoded bits is then mapped to a QPSK symbol. The variance of the noise corrupting this symbol is adjusted using the C/N of the location this symbol has been received. Remember that the C/N has been evaluated over a distance of 1 km with a spatial resolution of 0.004545 m. Assuming a bitrate of 250 kbps at the output of the application layer and a header overhead of 20%, the number of bits received in each 0.004545 m-long space interval depends of the constant speed v of the receiver (1.5 m/s, 15 m/s, and 31 m/s) and is given in Table III.

Speed (m/s)	1.5	15	31
Number of received bits	2727	273	132

TABLE III

NUMBER OF RECEIVED BITS IN AN SPACE INTERVAL OF 0.004545 M

At low speed, several MPEG2 packets are received in the same space interval. When the corresponding C/N is bad, several MPEG2 packets may thus be lost.

The bit error rate (BER) at the input and output of the turbo decoder have been represented on Figure 7. For all speeds, results have been averaged over 1 s. There is a high variation of the BER with the location of the receiver. The C/N varies by several dB between two points distant of few centimeters. The consequence is that the input BER for each turbo codeword is highly variable. It follows that, though the average input BER (about $6.3\ 10^{-2}$) is acceptable for the considered turbo code, the average output BER is relatively high (see Table II). As a result, many MPEG2 packets will be received without error and some will contain many errors. Figure 7 also shows that for high speeds, a smoother channel is seen by the receiver. This effect is similar to that obtained with the use of interleavers at the physical layer.

B. Scenario 1: Reference

In this scenario, the video is encoded at 250 kbps. Channel coding is only in charge of the turbo code at physical layer. The protocol stack is exactly that described in Section III. No erroneous packet is transmitted by MPE to the upper layers.

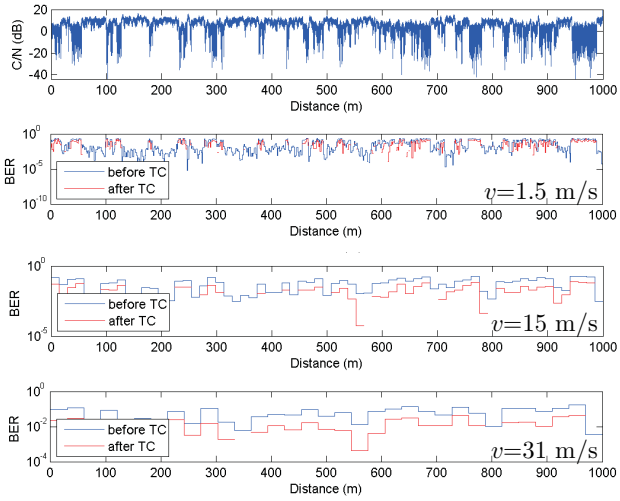

Fig. 7. BER before and after the turbo code for various speeds

Figure 8 presents the PSNR evaluated for the luminance of each received video frame. There is always a degradation of the PSNR when long shadowing occurs. Moreover, since there are

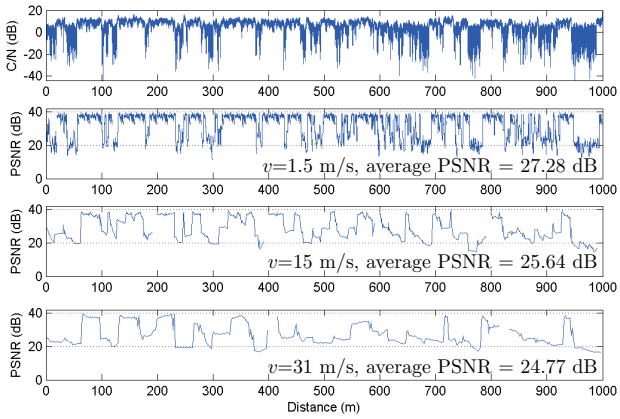

Fig. 8. Scenario 1, PSNR of each frame of the received video sequence

more discarded packets at high speed, error concealment has to be more intensively employed than at low speed. This has a direct impact on the PSNR of the reconstructed video sequences. It is generally admitted that the error concealment mechanisms of H.264 may compensate a PLR of 10%. The PLR of Table II are all larger than this limit, which explains the resulting bad but still acceptable video quality at $v = 1.5$ m/s in contrast with the unacceptable quality at

$v = 15$ m/s and $v = 31$ m/s.

Figure 9 represents the PSNR of the luminance of each decoded video frame as a function of the average C/N observed during the transmission of its corresponding compressed bistream. This illustrates that when the channel has a C/N above 5 dB, the received video is of good quality (PSNR larger than 32 dB).

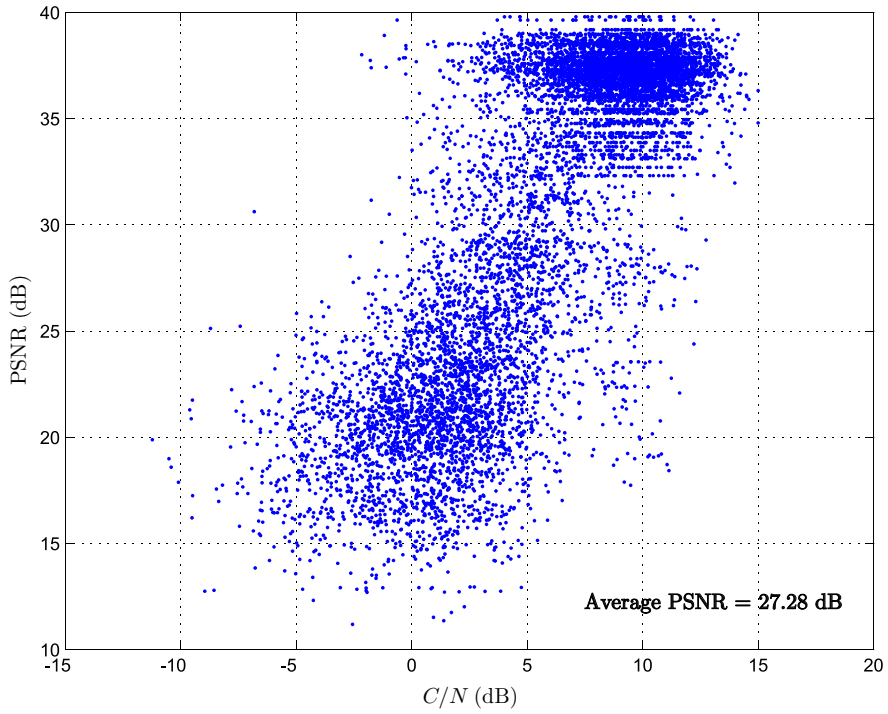

Fig. 9. PSNR of the reconstructed image as a function of the average C/N observed during its transmission for $v = 1.5$ m/s

In order to reduce the proportion of low quality frames, several solutions are investigated in the next sections. In the second scenario, presented in Section V-C, some spreading at the physical layer is introduced, in order to improve the efficiency of the turbo code. Next, one may think allowing some erroneously decoded packets to reach the upper layers, as suggested *e.g.* by [43] or UDP-lite, recently standardized [44], which modify the areas protected by the checksum to allow erroneous data to reach the application layer. Nevertheless, this option will not be presented here: experimentations have shown that the reconstructed video quality is not acceptable, the current implementations of the H.264 decoder being particularly prone to transmission errors. Robust video decoders taking into account source constraints have to be employed in this context [10]. The second scenario, described in Section V-D, thus considers a packet-erasure code, such

as that detailed in Section IV-B, to recover lost RTP packets. Section V-E illustrates a facet of the various cross-layer optimisation solutions for the redundancy introduction in the protocol stack and focuses on the distribution of redundancy between the intermediate and the application layers.

C. Scenario 2: Spreading at physical layer

The high potential of the spreading at physical layer is evident when considering the error-correction capabilities of the turbo-code. Indeed, with an average input BER of about 6.3×10^{-2}, a rate $1/3$ turbo code should perform well. If the bursts of errors evidenced in Scenario 1 can be transformed into quasi-independent errors, the turbo code will produce quasi error-free data.

It has been shown in Section IV-C that the diversity of the data in the space, when the receiver moves, is function of the product *spreading* $\times$ *speed*. Clearly, the speed is not a parameter that can be modified by the system. Additionally, the results provided in Section V-B show that the maximum speed (31 m/s) is not sufficient to ensure a satisfying level of diversity.

It follows that the only way to improve the diversity is to spread data at the physical layer. This is done by transmitting the data of a turbo codeword in different time slots, and possibly by interleaving several codewords. This solution is used, *e.g.*, by the UMTS physical layer.

This technique has been implemented in the simulator by spreading the data belonging to a codeword over a duration larger than its natural duration. For example, for the video rate $R = 250$ kbps, each turbo codeword (of 3×188 bytes) is sent in roughly 4.7 ms. Thus, a spreading of the codewords over a duration of $T_\mathrm{s} = 100$ ms is done by interleaving 21 codewords.

The influence of the spreading duration T_s on the PSNR is provided in Figure 10. In order to compare the received video quality at various speeds, the PSNR has been represented as a function of the product $T_\mathrm{s}v$ for $v = 1.5$ m/s and $v = 15$ m/s.

These results are compatible with the analysis of the interleaving performance provided in [42], which shows that after a given threshold for the interleaving time, the maximum performance of the coding system is reached. From Figure 10, one may observe that for $T_\mathrm{s}v \geqslant 9$ m, it is possible to reach a level of diversity such that the turbo code corrects almost all the errors, allowing a good reconstructed video quality. This distance is of the same order of magnitude as the correlation distance of the channel of 1.5 m (see Section II).

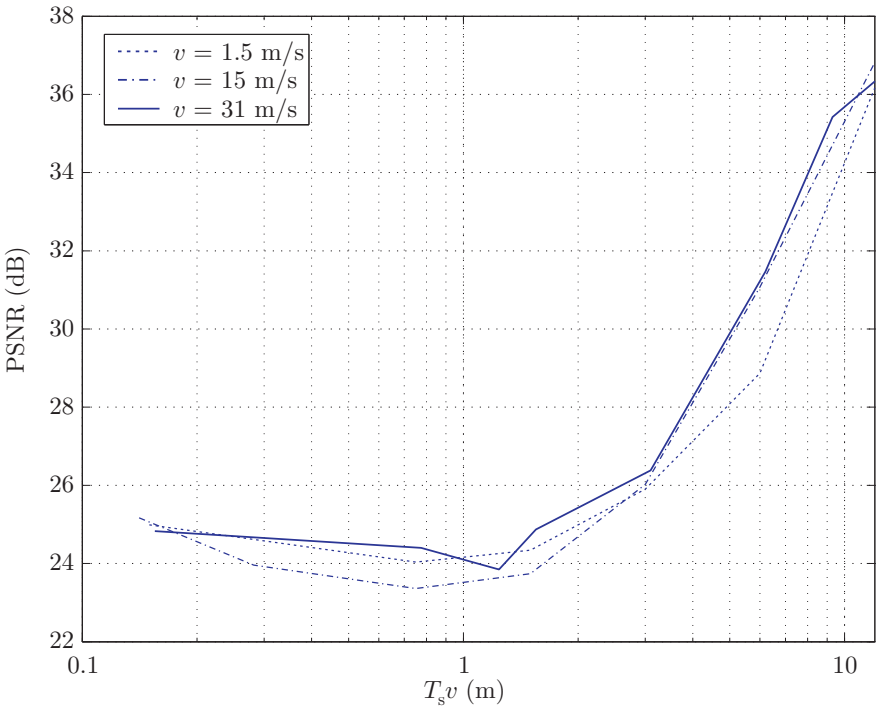

Fig. 10. Influence of the product $T_s v$ on the average PSNR of the reconstructed video sequence ($R = 250$ kbps)

Quasi error-free MPEG2 packets can be provided by the physical layer for all the speeds (including the pedestrian case) provided that $T_s v \geqslant 9$ m. Keeping in mind that the maximum zapping time is twice the spreading time, this leads to a good $T_s = 300$ ms for $v = 31$ m/s and an acceptable $T_s = 600$ ms for $v = 15$ m/s. Nevertheless, for $v = 1.5$ m/s, a spreading time $T_s \geqslant 6$ s would be required, which is not acceptable. Alternative solutions have thus to be considered.

D. Scenario 3: Packet erasure codes

This third scenario includes erasure codes between application and ROHC layers as described in Section IV. Rate $3/5$ Reed-Solomon erasure codes are employed and the rate of the coded video is reduced to $R = 150$ kbps, to get a total rate of 250 kbps after the packet erasure code, identical to the rate without erasure code in Scenarii 1 and 2. No spreading at physical layer has been considered in this scenario.

Figure 11 shows the influence of the erasure code by comparing the evolution of the PSNR for $T_{\text{block}} = 1,\ 2$ and 3 s for $v = 15$ m/s to the case without erasure code but with a video with

$R = 250$ kbps.

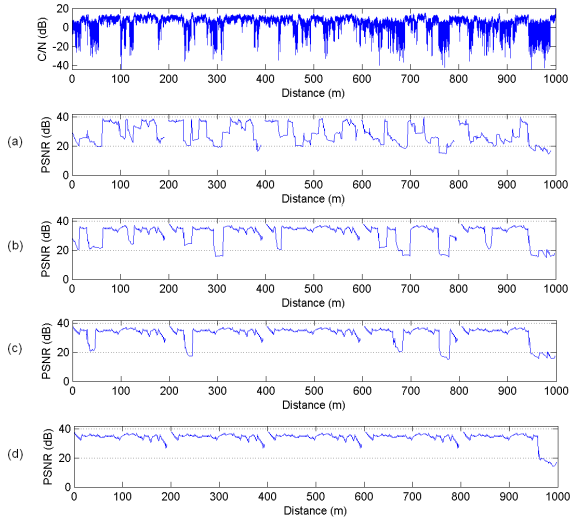

Fig. 11. PSNR for $v = 15$ m/s and for (a) $T_{\text{block}} = 0$ s (no erasure code, but with a video rate at 250 kbps), (b) $T_{\text{block}} = 1$ s, (c) $T_{\text{block}} = 2$ s and (d) $T_{\text{block}} = 3$ s,

The average PSNR and corresponding PLR for all combinations of T_{block} and v are given in Table IV. The characteristics of the code employed are also indicated. The reference case (without erasure code) is the first column.

T_{block}	0 s	1 s	2 s	3 s
Speed	- no FEC	RS$(261, 157)$	RS$(521, 313)$	RS$(782, 469)$
1.5 m/s	27.28 (0.12)	28.80 (0.07)	29.16 (0.07)	29.39 (0.07)
15 m/s	25.64 (0.17)	27.84 (0.10)	30.08 (0.06)	32.68 (0.02)
31 m/s	24.77 (0.18)	29.41 (0.09)	30.51 (0.05)	31.73 (0.04)

TABLE IV

AVERAGE PSNR (PLR) FOR $v = 1.5$, $v = 15$ AND $v = 31$ M/S AND FOR $T_{\text{BLOCK}} = 0$ (NO FEC), 1, 2 AND 3 SECONDS.

The erasure code clearly improves the performance of the system. Indeed, for all speeds, the performance with a video rate of 250 kbps is lower than that for a video rate of 150 kbps and an erasure code of rate $3/5$. Additionally, except for the speed equal to 1.5 m/s, the quality of

the video is significantly improved (about 2.5 dB for $v = 15$ m/s) each time the value of T_{block} is increased. This is confirmed by Figure 12 showing the cumulative distribution of the frame PSNR for various T_{block}. For $T_{\text{block}} = 1$ s, about 30% of the frames have a PSNR lower than 30 dB; for $T_{\text{block}} = 2$ s and $T_{\text{block}} = 3$ s, this proportion reduces to 17% and 8% respectively.

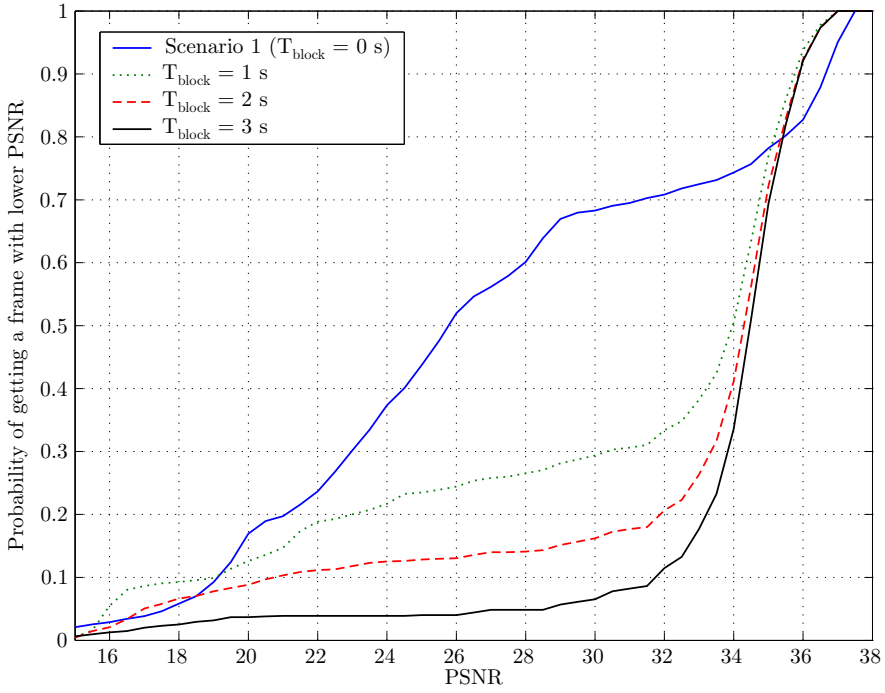

Fig. 12. Cumulative distribution of the frame PSNR for various T_{block}, $v = 15$ m/s

The better performance of the erasure code in this scenario may be explained by the bursty erasure patterns of the channel. Indeed, concealment techniques are efficient in the presence of randomly spaced erasures and much less in the presence of bursts of erasures. On the contrary, an erasure code is able to correct bursts of erasures provided that the burst length is lower than its correction capability.

The mean burst length explains the relatively poor performance of the erasure code for $v = 1.5$ m/s. Indeed, as explained in Section II, the channel considered in our simulations shows areas with different levels of reception. A mobile crossing an area with a bad reception level at 1.5 m/s observes a burst erasure length equal to 10 times the burst erasure length of a mobile crossing this area at 15 m/s. The simulations shows that bursts of erasures experimented by a receiver moving at 1.5 m/s are too long for the implemented erasure codes.

Even if the spreading at physical layer outperforms the solution proposed in this section for $v = 15$ m/s and $v = 31$ m/s, introducing redundancy at the intermediate layers turns out to provide more robustness at $v = 1.5$ m/s, while keeping an acceptable quality at higher speeds. Nevertheless, the required zapping times, $1.5T_{\text{block}}$ in average, between 1.5 s and 4.5 s are still large. Here again, robust source decoders may allow the use of a shorter T_{block} at low speeds.

E. Scenario 4: Cross-layer redundancy optimization

Section V-D shows that using erasure codes at intermediate layer significantly improves the received video quality. Further improvements may be obtained by optimizing the redundancy introduced at intermediate layer for a given T_{block}. Increasing the redundancy improves robustness, but requires a reduction of the rate R at the output of the video coder, leading to a reduction of the maximum expectable video quality.

New simulations have been performed with erasure codes with rates $2/5$, $3/5$, and $4/5$. The output rate R of the video coder has been adjusted accordingly, in order to keep the rate at the output of the packet erasure coder at 250 kbps. In this scenario, the size of the NALUs has been limited to 160 bytes.

	T_{block} (s)				
R	0	1	2	3	Noise-free
250 kbps	24.5				36.8
200 kbps	24.6	26.2 RS$(261, 157)$	26.5 RS$(521, 313)$	27.5 RS$(782, 469)$	35.8
150 kbps	23.9	28.2 RS$(196, 118)$	30.1 RS$(391, 235)$	30.8 RS$(586, 252)$	34.4
100 kbps	23.1	28.4 RS$(131, 79)$	30.1 RS$(261, 157)$	30.1 RS$(391, 233)$	32.3

TABLE V

AVERAGE PSNR (IN DB) FOR $v = 15$ M/S AND FOR VARIOUS COMBINATIONS OF THE VIDEO CODING RATE R AND OF T_{BLOCK}.

Table V summarizes the results obtained for $v = 15$ m/s. For small values of T_{block}, increasing the redundancy improves the reconstructed video quality. For larger values of T_{block}, it is better

to use less redundancy to get an increased video quality. A trade-off between redundancy introduction and compression has thus to be attained and this trade-off depends on the value T_{block}. At fixed redundancy, only an increase of T_{block} allows an increase of the nominal video quality.

VI. Conclusion

This paper compares several reliability mechanisms built around a protocol stack for a reliable satellite digital mobile video broadcast application. Several conclusions may be drawn from this study.

First, it was shown that the spreading/interleaving of the data on the physical layer obtained definitely the best performance in terms of packet loss rate. Provided that the turbo codewords are spread over a distance of more than 9 m, the turbo code obtains its best performance by generating a packet loss rate lower than 1%. Nevertheless, for a pedestrian moving at 1.5 m/s, this corresponds to the spreading of a codeword over a duration of 6 s, which results in an unacceptable average zapping time of 9 s. Another limitation of this solution is the increase of the hardware complexity. Indeed, the receiver has to store and process an amount of data corresponding to the spreading duration.

Second, it has been mentioned that allowing erroneous data to reach the application layer is not a viable solution. Indeed, even if the recent video standards such H.264 are able to cope with packets corrupted with few errors, the satellite channel is too bursty and the channel code generates a level of bit errors at the output of the physical layer that are not supported by the video decoder. Whether robust decoding of video can play a useful role in this situation is still an open question. This conclusion holds temporarily with the actual JM 10.2 decoder.

Third, Scenarii 1, 3, and 4 have shown that reducing the nominal video quality to allow some redundancy introduction at lower protocol layers is necessary to achieve satisfying performance. The error concealment mechanisms of H.264 have a performance limit, and may only account for a moderate proportion of lost packets. Erasure codes, evaluated in Scenarii 3 and 4, obtained good performance in all the simulated cases. Even if they have some drawbacks, such as the increase of the CPU and storage resources, they improved the final PSNR of the video for each speed and for each length of codes. Their main limitation is the zapping constraint which indirectly limits their length and thus their performance in terms of correction capability.

However, this mechanism offers several advantages. Indeed, their software implementation on higher layers allows a global flexibility (easy updates, reception of the data for terminals which do not implement the mechanism, easy activation/deactivation...) which can be of great interest in the SDMB system.

Extensions of this work include more cross-layer interactions, in order to provide richer information from the physical layer to the upper layer. This may allow to use error-correcting codes at intermediate layers in place of erasure codes. Soft decoding of the video taking into account the redundancy left by the video coder, as illustrated in [10], may also significantly increase the performance. Finally, the use of the scalable extension of H.264 [45], in order to be able to broadcast video to receivers with various display and computing capabilities, using a single video stream may also be interesting.

This paper has studied a set of solutions to improve the reliability in satellite-mobile transmissions. These techniques were evaluated and their strengths, the drawbacks and their relations with the constraints of the system were presented. It seems difficult to advocate a technique related to the others since their use strongly depends on the choices made on the system at the different layers. However, the results presented in this paper should be useful for the designers of such systems.

REFERENCES

[1] ETSI, "Multimedia Broadcast/Multicast service. (MBMS); UTRAN/GERAN requirements," 3GPP TR 25.992-140, Tech. Rep., jun. 2005.

[2] ——, "Digital audio broadcasting (DAB); DMB video service; user application specification," ETSI TS 102 428 V1.1.1, Tech. Rep., jun. 2005.

[3] ——, "Digital video broadcasting (DVB); transmission system for handheld terminals (DVB-h)," ETSI EN 302 304 v1.1.1, Tech. Rep., nov. 2004.

[4] T. Stockhammer, M. M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environment," *IEEE trans. on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 657–673, 2003.

[5] F. Pérez Fontán, M. Vázquez-Castro, C. E. Cabado, J. P. García, and E. Kubista, "Statistical modelling of the lms channel," *IEEE Transactions On Vehicular Technology*, vol. 50, no. 6, pp. 1549–1567, november 2001.

[6] ITU-T and ISO/IEC JTC 1, "Advanced video coding for generic audiovisual services," ITU-T Rec. H.264, and ISO/IEC 14496-10 AVC, Tech. Rep., nov. 2003.

[7] M. Van der Schaar and S. Shankar, "Cross-layer wireless multimedia transmission: Challenges, principles, and new paradigms," *IEEE Wireless Communications Magazine*, vol. 12, no. 4, pp. 50–58, 2005.

[8] H. Jenkac, T. Stockhammer, and W. Xu, "Cross-Layer Assisted Reliability Design for Wireless Multimedia Broadcast," *EURASIP Signal Processing Journal, Special Issue on Advances in Signal Processing-assisted Cross-layer Designs*, 2006, accepted for Publication.

[9] H. Nguyen and P. Duhamel, "Compressed image and video redundancy for joint source-channel decoding," in *Proc. Globecom*, 2003.

[10] G. Sabeva, S. Ben-Jamaa, M. . Kieffer, and P. Duhamel, "Robust decoding of h.264 encoded video transmitted over wireless channels," in *Proceedings of MMSP*, Victoria, Canada, 2006.

[11] ITU-R, *Propagation data and prediction methods required for the design of Earth-space telecommunication systems*, ITU-R P.618.

[12] C. Loo, "A statistical model for land mobile satellite link," *IEEE Trans. Veh. Technol.*, vol. 34, pp. 122–127, august 1985.

[13] C. Bazile, B. Martin, G. Scot, O. Courseille, F. Lacoste, and C. Loisel, "Status on satellite mobile tv air interface issues," in *3rd Advanced Satellite Mobile Systems Conference, ASMS'06 conference*, may 2006.

[14] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, 1996, Request for Comments 1889.

[15] S. Wenger, M. Hannuksela, T. Stockhammer, and M. Westerlund, *RTP Payload Format for H.264 Video*, February 2005, Request for Comments 3984.

[16] DVB (Digital Video Broadcasting, "IP Datacast over DVB-H: Architecture," November 2005.

[17] J. Postel, *User Datagram Protocol*, 1980, Request for Comments 768.

[18] ——, *Internet Protocol*, Request for Comments 760.

[19] S. Deering and R. Hinden, *Internet Protocol, Version 6 (IPv6) Specification*, december 1998, Request for Comments 2460.

[20] C. Bormann and al., *RObust Header Compression (ROHC):Framework and four profiles: RTP, UDP, ESP, and uncompressed*, July 2001, Request for Comments: 3095, Standards Track .

[21] A. Couvreur, L.-M. Le-Ny, A. Minaburo, G. Rubino, B. Sericola, and L. Toutain, "Performance analysis of a header compression protocol: The rohc unidirectional mode," *Telecommunication Systems*, vol. 31, pp. 85–98, 2006.

[22] ISO/IEC JTC 1/SC 29, "Generic coding of moving pictures and associated audio information: Systems," ISO/IEC 13818-1 (MPEG-2 Part 1), Tech. Rep., 1996.

[23] G. Fairhurst and B. Collini-Nocker, *Unidirectional Lightweight Encapsulation (ULE) for Transmission of IP Datagrams over an MPEG-2 Transport Stream (TS)*, 2005, Request for Comments 4326.

[24] *Digital Video Broadcasting (DVB); DVB specification for data broadcasting*, 2004, european Telecommunications Standards Institute, EN 301 192.

[25] ETSI, "Digital video broadcasting (DVB); framing structure, channel coding and modulation for digital terrestrial television," ETSI EN 300 744 v1.5.1, Tech. Rep., jun. 2004.

[26] ——, "Universal mobile telecommunications system (UMTS); multiplexing and channel coding (FDD) (3GPP TS 25.212 version 6.7.0 release 6)," ETSI TS 125 212 V6.7.0, Tech. Rep., dec. 2005.

[27] ——, "Physical layer standard for cdma2000 spread spectrum systems," 3GPP2 C.S0002-D version 1, Tech. Rep., feb. 2004.

[28] R. G. Gallager, *Low Density Parity Check Codes*. Cambridge: MIT Press, 1963.

[29] ETSI, "Digital video broadcasting (DVB); second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications," ETSI EN 302 307 V1.1.1, Tech. Rep., mar. 2005.

[30] A. Matache, S. Dolinar, and F. Pollara, "Stopping rules for turbo decoders," Tech. Rep., August 2000, TMO Progress Report 42-142.

[31] M. Van der Schaar, S. Krishnamachari, S. Choi, and X. Xu, "Adaptive cross-layer protection strategies for robust scalable video transmission over 802.11 WLANs," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 10, pp. 1752–1763, 2003.

[32] Y. Wang and Q. Zhu, "Error control and concealment for video communication: A review," *Proceedings of the IEEE*, vol. 86, pp. 974–997, 1998.

[33] Y. Wang, S. Wenger, J. Wen, and A. Katsaggelos, "Review of error resilient coding techniques for real-time video communications," *IEEE Signal Proc. Magazine*, vol. 17, no. 4, pp. 61–82, 2000.

[34] Q.-F. Zhu, Y. Wang, and L. Shaw, "Coding and cell loss recovery in DCT based packet video," *IEEE Trans. Circuits and Systems for Video Technology, special issue on Packet Video*, vol. 3, no. 3, pp. 248–258, 1993.

[35] W. M. Lam, A. R. Reibman, and B. Liu, "Recovery of lost or erroneously received motion vectors," in *Proceedings ICASSP*, vol. 5, 1993, pp. 417–420.

[36] M. C. Hong, H. Schwab, L. P. Kondi, and A. K. Katsaggelos, "Error concealment algorithms for compressed video," *Signal Processing: Image Communication*, vol. 14, pp. 473–492, 1999.

[37] M. W. M. Luby, A. Shokrollahi and T. Stockhammer, *Raptor Forward Error Correction Scheme for Object Delivery, draft-ietf-rmt-bb-fec-raptor-object-04*, 2006, work in progress, draft-ietf-rmt-bb-fec-raptor-object-04, Internet draft.

[38] V. Roca, Z. Khallouf, and J. Laboure, "Design and evaluation of a low density generator matrix (ldgm) large block fec codec," in *Fifth International Workshop on Networked Group Communication (NGC'03), Munich, Germany*, September 2003.

[39] J. Lacan, V. Roca, J. Peltotalo, and S. Peltotalo, *Reed-Solomon Forward Error Correction (FEC), draft-ietf-rmt-bb-fec-rs-01*, 2006, work in progress, draft-ietf-rmt-bb-fec-rs-01, Internet draft.

[40] Nokia, *Simulation results for the performance and complexity of RS Codes for MBMS FEC*, 3GPP TSG SA WG4 PSM -7 Tdoc S4-AHP221, Sophia Antipolis, France, 6-8 April 2005.

[41] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data." in *SIGCOMM*, 1998, pp. 56–67.

[42] M. Zorzi and R. R. Rao, "On the statistics of block errors in bursty channels," *IEEE Transactions on Communications, Vol.45, No. 6*, pp. 660–667, June 1997.

[43] E. Masala, M. Bottero, and J. De Martin, "MAC-level partial checksum for H.264 video transmission over 802.11 ad hoc wireless networks," in *Proc. IEEE 61st Vehicular Technology Conference*, May 2005.

[44] L.-A. Larzon, M. Degermark, S. Pink, L.-E. Jonsson, and G. Fairhurst, *The Lightweight User Datagram Protocol (UDP-Lite)*, 2002, Request for Comments 3450, Standards Track.

[45] H. Schwartz, D. Marpe, and T. Wiegand, "Scalable extension of h.264/AVC," ISO/IEC JTC 1/SC29/WG11 MPEG 2004/M10569/S03, München, Tech. Rep., mar. 2004.